Securing Network Automation

Ivan Pepelnjak (ip@ipSpace.net)
Network Architect

ipSpace.net AG

Who is Ivan Pepelnjak (@ioshints)

Past

- Kernel programmer, network OS and web developer
- Sysadmin, database admin, network engineer, CCIE
- Trainer, course developer, curriculum architect
- Team lead, CTO, business owner

Present

- Network architect, consultant, blogger, webinar and book author

Focus

- SDN and network automation
- Large-scale data centers, clouds and network virtualization
- Scalable application design
- Core IP routing/MPLS, IPv6, VPN

CISCO CCIE EMERITUS

vmware vEXPERT

What’s In It For Me
(Why Should I Automate)

Sounds Familiar?

- Increase flexibility while reducing costs
- Faster application deployments
- Compete with public cloud offerings

What Would You Automate?

Every Well-Defined Repeatable Task Can Be Automated

What Would You Automate?

Common answers:

- Device provisioning
- Service provisioning (= device configurations)
- VLANs
- ACLs
- Firewall rules

How about…

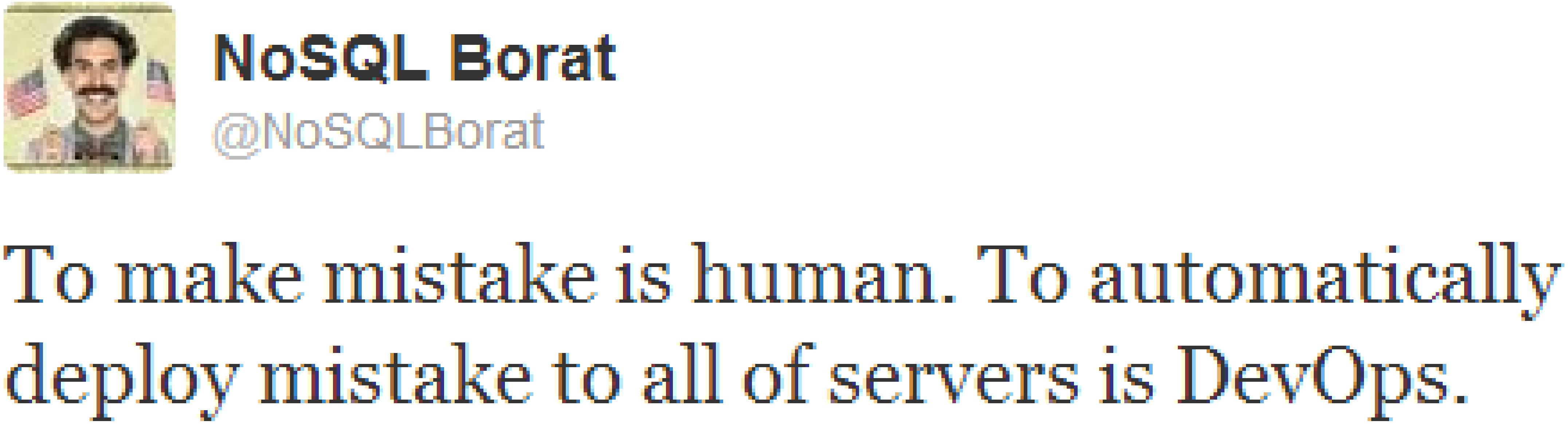
- Troubleshooting
- Consistency checks
- Routing adjustments
- Failure remediation

Build or Buy?

You’ll Have to Build Anyway

The Interesting Questions

- What do I need?
- How soon do I need it?
- Can I buy what I need?
- How much will that cost?
- How much customization will that require?
- How locked-in will I be?
- How extensible is the product I’m considering?
- Do I have the resources to build it?
- Do I have internal (management) support to build it?
- Can I start small?
- Can I get help (master builders)?
- How long will it take to build it?



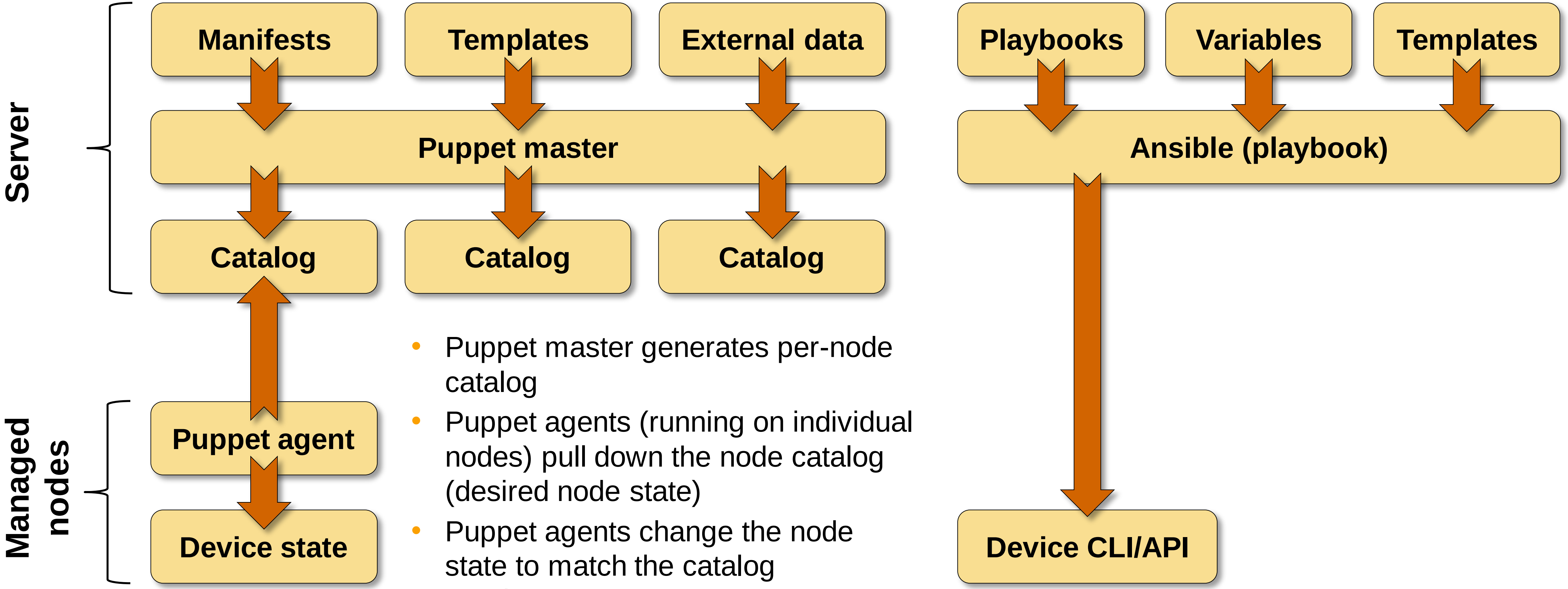
NoSQL Borat

@NoSQLBorat

To make mistake is human. To automatically deploy mistake to all of servers is DevOps.

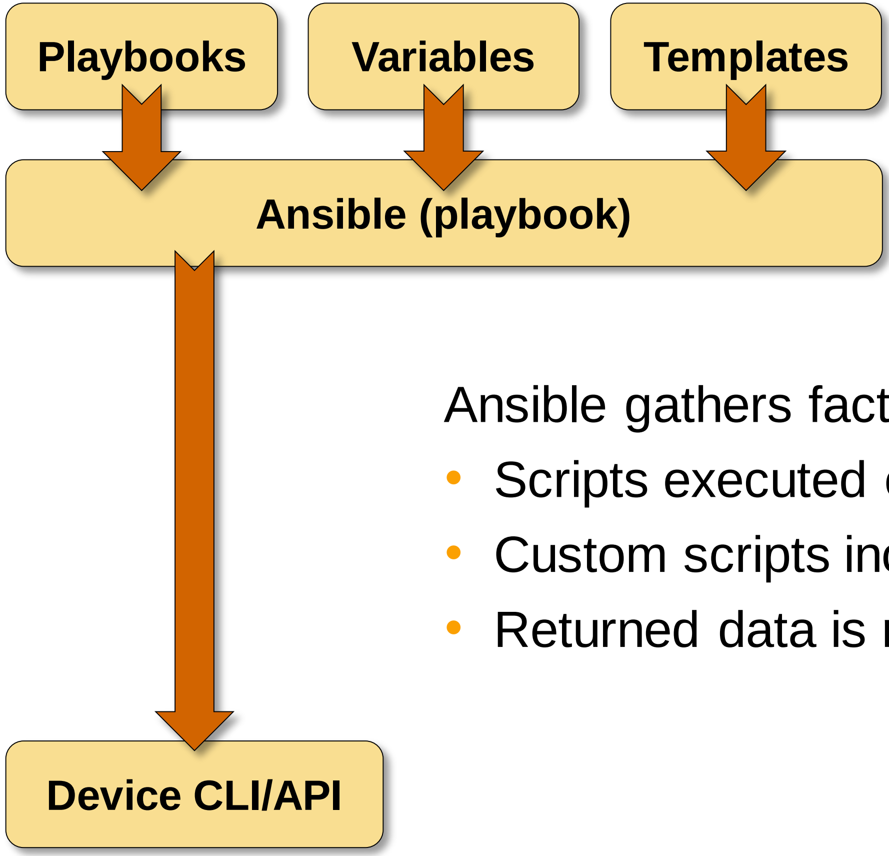
Security Aspects

Example: Puppet or Chef versus Ansible



- Puppet master generates per-node catalog
- Puppet agents (running on individual nodes) pull down the node catalog (desired node state)
- Puppet agents change the node state to match the catalog requirements

Sidetrack: Ansible Vulnerabilities



Ansible gathers facts from managed devices

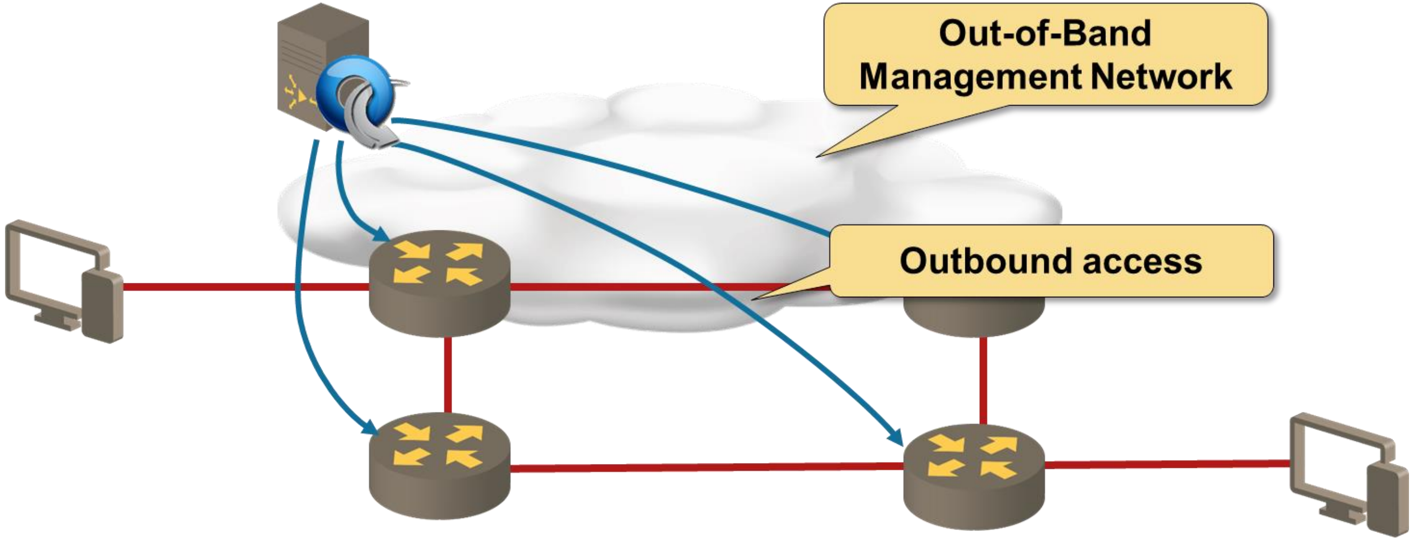
- Scripts executed on managed devices ➔ data injection opportunity
- Custom scripts included in fact gathering ➔ more data injection
- Returned data is not properly quoted/parsed ➔ privilege escalation

Not applicable to most network devices (no fact gathering, no custom scripts)

Solutions

The Usual

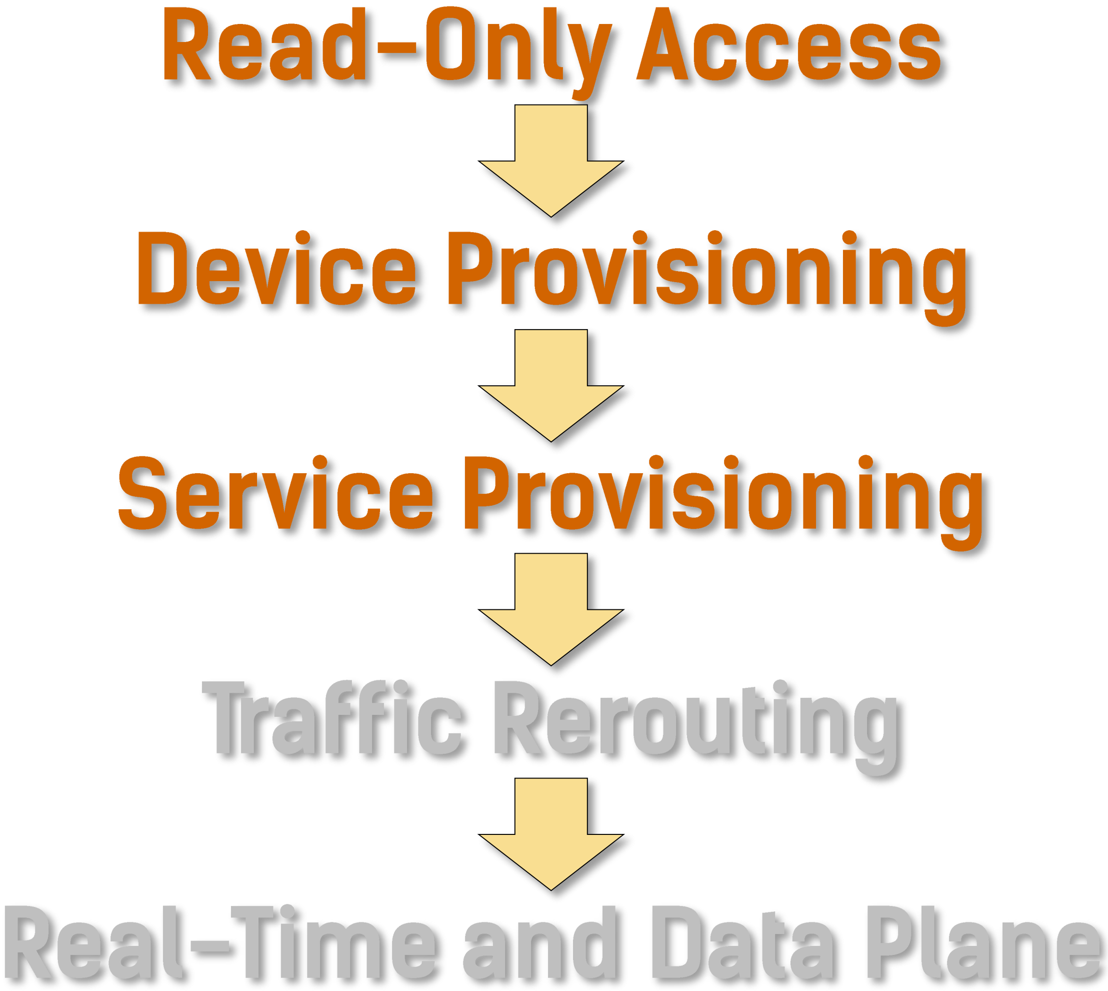
- Out-of-band management
- Management network/VRF
- Limit access to management hosts
- SSH-based access
- Use SSH keys
- Role-based access control (commit scripts)



Out-of-Band Management Network

Outbound access

No different from traditional network management systems



Read-Only Access

Device Provisioning

Service Provisioning

Traffic Rerouting

Real-Time and Data Plane

This is how you start

Reliability Aspects

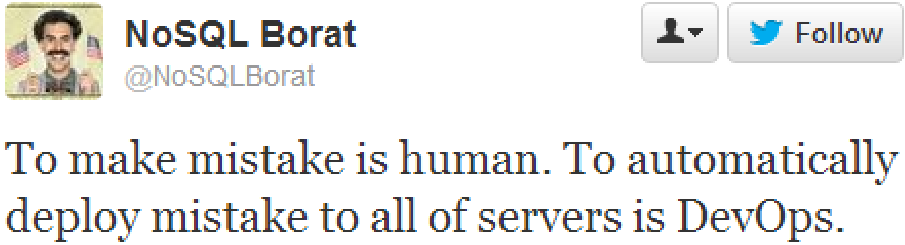
Shall We Program the Network?

Keep in mind

- Network is your most critical infrastructure
- Treat network programming like any other critical application

You need

- Programming skills
- Deep understanding of the desired network behavior
- Tools, processes and procedures
- Test environment and QA
- Deployment procedures



NoSQL Borat
@NoSQLBorat

To make mistake is human. To automatically deploy mistake to all of servers is DevOps.

Applies equally well to home-grown automation or vendor SDN solution

Principles

Trust is good but control is better

- Don’t trust input data
- Don’t trust device state
- Assert your assumptions
- Fail on unexpected results (device-supported rollback helps)

Validate successful deployment

- Execute **show** commands after configuration change
- Check actual device state, neighbors…
- Fail (or report error) on mismatch

Test, Test, Test … and Test Some More

Unit tests

- Test every single component with valid and all possible invalid inputs

Functional/integration tests

- Does the automation solution generate the desired configurations?
- Use mockups (check executed commands, return pre-collected printouts)

Continuous Integration

- Generate a test lab and execute tests after every committed change
- Virtual lab for quick checks, physical gear for pre-deployment tests
- Your vendor doesn’t want to give you device VMs? Change the vendor!

Post-Deployment Tests

Compare actual and expected network state

- HSRP/VRRP/OSPF/BGP/EIGRP neighbors
- Number of prefixes received from each neighbor
- Traffic statistics (need baseline and anomaly detector)

Perform connectivity tests

- Is the traffic flowing where I expect it to flow?
- Are ACLs or firewall rules working as expected?

Use post-deployment tests for continuous network validation

Gaining the Trust

Read-only access

- Non-intrusive solutions that add immediate value
- API access or topology collection/extraction (example: BGP)
- Leverage end-to-end visibility (usually ignored by NMS)

Configuration generation (templates)

- Cut-and-paste
- Verify-and-deploy (use **check** mode with Ansible)
- Automatic deploys in maintenance windows
- Automatic real-time deploys

More extensive programming

- Control-plane interactions (BGP, RTBH, BGP FlowSpec)
- Read-write API access (example: DirectFlow)

Hint: Get management buy-in and professional programmers

Takeaway

You'll Be Developing Software No Matter What

Get used to it

- The only way to get agile is to automate deployments
- The only way to automate deployments is to buy or build automation solutions
- Don't trust vendors (or their solutions)
- You don't have to become programmer
- You **MUST** think about **SYSTEMS** and **PROCESSES**

> The real tiger is never a match for the paper one, unless actual use is wanted.

Mythical Man-Month (Frederick P. Brooks, 1975)

You'll Be Developing Software No Matter What

Getting there

- Build a prototype to prove the concept
- Get management buy-in
- Get senior software developer(s) in your team
- Get a few programmers
- Cross-pollinate ;)

> In most projects, the first system built is barely usable
>
> The only question is whether to plan in advance to build a throwaway, or to promise to deliver the throwaway to customers.

Mythical Man-Month (Frederick P. Brooks, 1975)

Gartner on Shiny New Object Syndrome

[…] address the following questions before introducing any new technology:

- Can the root issue be addressed via a policy or process change?
- If we wait a year, will this become a commoditized capability from established providers (or my existing providers)?
- Do we have existing network, security, or management capabilities that can address the bulk (i.e., 85%) of the technological requirements?
- Do we have the right process and staff expertise to properly leverage the new technology?

Source: http://blogs.gartner.com/andrew-lerner/2015/01/15/netsecdirtydozen/

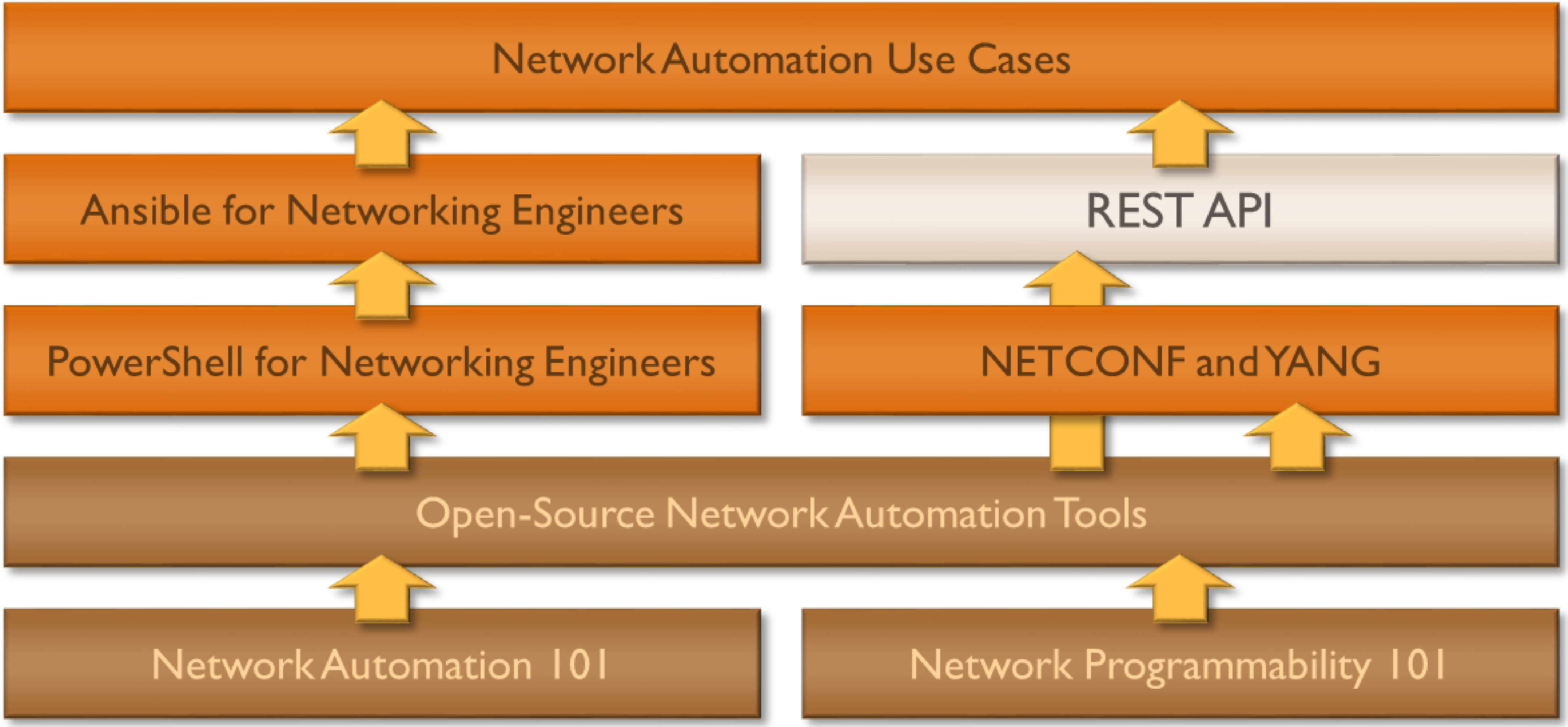
Vote with Your Wallet

What Should You Ask For?

- Programmable interface (API)
- Structured operational data (in JSON or XML format)
- Device configuration in structured (JSON/XML) format
- Atomic configuration changes (candidate configuration + commit/rollback)
- Configuration rollback
- Configuration replace
- Contextual configuration diff
- Support for industry-standard models (IETF and OpenConfig)
- Feature parity (API to CLI)

More @ http://blog.ipspace.net/2016/10/network-automation-rfp-requirements.html

Network Automation Track


Network Automation Use Cases
Ansible for Networking Engineers
REST API
PowerShell for Networking Engineers
NETCONF and YANG
Open-Source Network Automation Tools
Network Automation 101
Network Programmability 101

More information @ ipSpace.net/automation

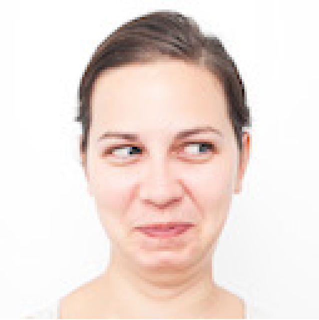
Building Network Automation Solutions

6 week advanced interactive online course

Course starting in

September 2017

- High-intensity online course
- Hands-on experience developing automation solutions
- 6-week course spread across 2 months
- Live online discussions and guest speaker sessions
- Design and coding assignments



More information @ automation.ipSpace.net

Questions?

Send them to ip@ipSpace.net or @ioshints